Amendments to the claims,
Listing of all claims pursuant to 37 CFR 1.121(c)

This listing of claims will replace all prior versions, and listings, of claims in the application:

What is claimed is:

1. (Currently amended) In a database system, a self-tuning method for performing recovery operations ~~using an optimal number of~~ by dynamically adapting how many recovery threads are spawned during recovery, the method comprising:

(a) spawning an initial recovery thread to perform recovery operations;

(b) measuring I/O (input/output) performance with the initial recovery thread;

(c) spawning a subsequent recovery thread to perform recovery operations;

(d) measuring I/O performance with the subsequent recovery thread; and

(e) self-tuning how many threads are spawned by continuing, as long as I/O performance does not degrade beyond a preselected percentage, ~~repeating~~ to repeat steps (c) and (d) for spawning a desired number of additional recovery threads.

2. (Original) The method of claim 1, wherein I/O performance is measured over a given period of time.

3. (Original) The method of claim 2, wherein the given period of time is about 1 second.

4. (Original) The method of claim 1, wherein steps (c) and (d) are repeated for spawning additional recovery threads, as long as I/O performance degrades by no more than about 15 percent.

5. (Original) The method of claim 1, wherein steps (c) and (d) are repeated such that only a preconfigured maximum number of recovery threads may be generated.

6. (Original) The method of claim 5, wherein the maximum number of recovery threads is limited to not exceed a count of databases that can be opened.

7. (Original) The method of claim 5, wherein the maximum number of recovery threads is limited to not exceed one less than a count of database engines online.

8. (Currently amended) The method of claim 1, wherein step (e) further comprises:

when I/O performance measured for a just-spawned recovery thread degrades beyond the preselected percentage, putting the just-spawned recovery thread to sleep.

9. (Original) The method of claim 8, further comprising:

after another recovery thread finishes, awaking the thread that has been put to sleep.

10. (Original) The method of claim 1, wherein steps (c) and (d) are repeated up to a configured maximum number of databases that can be recovered concurrently.

11. (Original) The method of claim 1, wherein each recovery thread itself recovers a single database at a time.

12. (Currently amended) The method of claim 1, wherein a user of the system is able to specify a particular number of concurrent recovery threads, and wherein the system generates an advisory if the particular number of concurrent recovery threads specified ~~is not optimal~~ can be changed to achieve better I/O performance.

13. (Original) A computer-readable medium having processor-executable instructions for performing the method of claim 1.

14. (Currently amended) The method of claim 1, further comprising:

~~A downloadable~~ downloading a set of processor-executable instructions for performing the method of claim 1.

15. (Currently amended) A database system performing self-tuning recovery operations ~~using an optimal number of~~ by dynamically adapting how many recovery threads are spawned during recovery, the system comprising:

a database system having at least one database that may require recovery;

an initial recovery thread that is spawned to perform recovery operations, wherein the system measures I/O (input/output) performance with the initial recovery thread; and

a plurality of additional recovery threads that are spawned to perform recovery operations, wherein the system dynamically adjusts how many recovery threads are spawned based on ~~measures~~ I/O (input/output) performance with each additional recovery thread that is spawned, and wherein the system ceases spawning additional recovery threads when I/O performance degrades beyond a desired amount.

16. (Original) The system of claim 15, wherein I/O performance is measured over a given period of time.

17. (Original) The system of claim 16, wherein the given period of time is about 1 second.

18. (Original) The system of claim 15, wherein the system may spawn additional recovery threads as long as I/O performance degrades by no more than about 15 percent.

19. (Original) The system of claim 15, wherein the plurality of additional recovery threads spawned is limited such that only a maximum number of recovery threads may be generated.

20. (Original) The system of claim 19, wherein the maximum number of recovery threads is limited to not exceed a count of databases to be opened.

21. (Original) The system of claim 19, wherein the maximum number of recovery threads is limited to not exceed one less than a count of database engines online.

22. (Currently amended) The system of claim 15, wherein, when I/O performance measured for a just-spawned recovery thread degrades beyond the desired amount, the system puts the just-spawned recovery thread to sleep.

23. (Original) The system of claim 22, wherein the system awakens the thread that has been put to sleep, upon termination of another thread.

24. (Original) The system of claim 15, wherein a maximum number of recovery threads permitted is limited to a configured maximum number of databases that can be recovered concurrently.

25. (Original) The system of claim 15, wherein each recovery thread recovers a particular database at a given point in time.

26. (Currently amended) The system of claim 15, wherein a user of the system is able to specify a particular number of concurrent recovery threads, and wherein the system generates an advisory if the particular number of concurrent recovery threads specified ~~is not optimal~~ can be changed to achieve better I/O performance.

27. (Currently amended) In a database system, an auto-tuning method for performing database recovery in a manner that dynamically adjusts how many recovery threads are spawned based on current performance, the method comprising:

spawning a thread to perform database recovery and recording statistics about performance associated with that thread; and

during recovery, dynamically adjusting how many threads are spawned by performing substeps of:

attempting to spawn additional threads to perform database recovery and recording statistics about performance associated with each additional thread spawned; and

if the performance for a given thread degrades beyond a desired amount,

freezing execution of the given thread and ceasing any attempt to spawn additional threads for database recovery.

28. (Original) The method of claim 27, wherein performance comprises I/O (input/output) performance measured over a given period of time.

29. (Original) The method of claim 28, wherein the given period of time is about 1 second.

30. (Original) The method of claim 27, wherein the desired amount is no more than about 15 percent degradation in performance.

31. (Original) The method of claim 27, wherein only a certain maximum number of threads may be generated for performing database recovery.

32. (Original) The method of claim 31, wherein the maximum number of threads is limited to not exceed a count of databases that may be opened.

33. (Original) The method of claim 31, wherein the maximum number of threads is limited to not exceed one less than a count of database engines online.

34. (Currently amended) The method of claim 27, wherein the freezing step comprises:

when I/O performance of the system degrades beyond a preselected percentage, freezing the given thread.

35. (Original) The method of claim 27, further comprising:

after another one of the threads finishes, thawing the thread that has been frozen.

36. (Original) The method of claim 27, wherein the system does not attempt to spawn more threads than a maximum number of databases available to be recovered

concurrently.

37. (Original) The method of claim 27, wherein each thread recovers a particular database at a time.

38. (Currently amended) The method of claim 27, wherein a user of the system is able to specify a particular number of concurrent threads, and wherein the system generates an advisory if the particular number of concurrent threads specified ~~is not optimal~~ can be changed to achieve better I/O performance.

39. (Original) A computer-readable medium having processor-executable instructions for performing the method of claim 27.

40. (Currently amended) The method of claim 27, further comprising:

~~A downloadable~~ downloading a set of processor-executable instructions for performing the method of claim 27.

41. - 59. (Canceled)